INTERFACES
Vol. 00, No. 0, Xxxxx 0000, pp. 000–000
ISSN 0092-2102 | EISSN 1526-551X | 00 | 0000 | 0001

INFORMS
DOI 10.1287/xxxx.0000.0000

Lessons Learned from 15 Years of Operations Research for French TV Channel TF1

Thierry Benoist, Frédéric Gardi, Antoine Jeanjean
Bouygues e-lab, Paris, France
{tbenoist@bouygues.com, fgardi@bouygues.com, ajeanjean@bouygues.com}

Bouygues' corporate operations research team (the Bouygues e-lab) has been working with various subsidiaries of the company's industrial group, including the leading French TV channel TF1, for the past 15 years. This article presents our achievements in working with this advertising broadcaster and identifies the practical keys to success in this partnership. In particular, it points out and explains best practices for managing operations research projects. The selected projects, which cover the Internet and television businesses, gave TF1 a competitive advantage by allowing it to provide quicker and better answers to advertisers' requests and to better use its limited and perishable airtime inventory. TF1 estimates the resulting revenue increase at €20 million per year.

Key words: media and advertisement; sales and marketing; revenue management; OR project management.
History: This paper has been refereed.

Created in 1952, Bouygues is a diversified industrial group whose businesses focus on two sectors: construction and telecommunications and media. Construction includes Bouygues Construction (building, civil works, energy, and services), Bouygues Immobilier (property), and Colas (roads); telecommunications and media include TF1 and Bouygues Telecom. In 2010, Bouygues had over 133,000 employees in 80 countries, generating €31.2 billion in revenue. Bouygues' corporate operations research (OR) team (the e-lab) has been working for 15 years with the various subsidiaries of this group. This small team (five OR engineers) reports directly to Bouygues' chief information officer and acts as an analytics services company within the group. In particular, a long-term partnership with TF1 led to a series of achievements in the advertising business. The examples discussed in this paper are taken from work with TF1.

As background, TF1 is a major media group in Europe and the leading television (TV) group in France, with an almost 25 percent audience share and 45 percent of the TV advertising market. It operates 16 TV channels, 128 radio stations, and 15 websites, and generated €2.4 billion in sales in 2009 (including €1.4 billion from the TF1 core channel). TF1's advertising arm sells time to

advertisers. The TV advertisements (ads) it sells respect France's legal limit of 12 minutes per hour and 144 minutes per day. Ad slots are marketed according to customer segments targeted by advertisers (e.g., women under age 50, men aged 15 to 49) and the estimated viewing audience. TV viewership is expressed as gross rating point (GRP). GRP measures the audience reached by an advertising campaign: 16 GRP represents 16 percent of the audience target. For example, France has 14 million males aged 15 to 49; hence, 16 GRP for this marketing target represents 2.2 million viewers. TV advertisers buy a specific number of seconds during a commercial break. The price per second of such a slot depends on the expected viewing audience, the time of the day, and the advertiser's negotiated discount rate. Such slots can be bought either as one slot or as a package whose scheduling is computed by TF1 as a marketing planning service. The interested reader can find further details related to the TV advertising market in Gabszewicz et al. (1999) or in Goettler and Shachar (2001). Internet advertising is sold somewhat differently. The audience is often measured in terms of the number of visitors per day; the ability for a visitor to click on advertisements on a web page led to two different payment mechanisms: pay-per-view and pay-per-click. In the first model, the advertising company pays a fixed fee each time its message is displayed to an Internet visitor; in the pay-per-click model, the payment is proportional to the number of times a visitor clicks on the advertisement.

In this article, we present three optimization problems encountered in the advertising business. First, we illustrate our mathematical consulting activities with an analysis of Internet click rates in a pay-per-click context, leading to a bandit-arm algorithm. Then, we describe a goal programming model, which the sales front office uses each day to build high-quality marketing plans for TV advertisers. Finally, we present the global optimization problem for the group's main channel: an objective function that runs into the hundreds of millions of euros to be optimized every other month. Throughout the description of these projects, we will attempt to identify the keys to success; we eventually draw conclusions on the management of OR projects and on the importance of information technology (IT) issues in this context.

Mathematical Consulting: Optimizing Internet Click Rates

Business Context

Various mathematical questions arise in the advertising business. For example, when advertising on Internet sites, the probability that a user will click on a given advertisement is a crucial figure. On TF1 websites, which attract 17 million unique visitors per month, the so-called shopping-boxes follow this pay-per-click model. Shopping-boxes are advertising messages that present a

product rather than a brand, and include a direct link to a page where the Internet user can buy the product. Each advertising company provides a catalog of products and TF1 can freely select displayed products from this catalog. The number of products is close to 20,000. Each time a page is displayed, the advertisement server should select the product with the highest click probability. Because only clicks generate revenue, this advertisement selection algorithm must be carefully designed.

Optimization and Difficulty

A statistical analysis of historical data allows key factors impacting the click rate of an advertisement for the product to be identified. These factors include the day of the week (e.g., Friday), the time slot (e.g., from 5:00 pm to 6:59 pm), the product category (e.g., travel), the price level (e.g., €25 to €50), and the Internet site section (e.g., news, sports, cooking). A simple advertisement selection algorithm can be based on these facts. When selecting a product to display on a web page at a certain time, the advertisement server can use such data to pick a product with the appropriate price and category. However, the product's attractiveness appears to be the most important criterion; similar products can have dramatically different click rates; the product might be better or better advertised. For example, we must learn dynamically whether the blue swimsuit has a higher click rate than the red swimsuit.

Solution Technique

This choice of solution technique will be based on the observed clicking frequency of each product; however, the accuracy of these figures depends on the number of times a product was displayed. Here, we face a classic exploitation versus exploration dilemma: shall we display the product having the highest observed click rate or a product that was presented infrequently to see if its attractiveness is higher? In the literature, this problem is known as the multiarmed bandit problem. Given a number of bandit arms with unknown reward probabilities, how do we maximize the cumulative reward obtained with a fixed number of coins? A powerful strategy for this type of game consists in using the upper confidence bound (UCB) (i.e., the upper bound of a confidence interval around the observed click frequency), which Auer et al. (2002) introduced. Products presenting a high UCB are either those that show a high click frequency over a large number of displays (i.e., narrow confidence interval around a high click frequency) or those rarely presented yet (i.e., wide confidence interval possibly around a poor click frequency). That is, this criterion judiciously combines the exploitation and exploration aspects. Note that Pandey and Olston (2007) report significant gains by using this kind of strategy on Yahoo! query logs. The work of Nakamura and Abe (2005) discusses the use of linear programming at NEC Corporation in this context.

Front-Office Software: Optimizing Advertisers' Plans

Business Context

In addition to its general interest channel, TF1 also operates theme channels, which have become key components of the satellite, broadband, digital terrestrial television (DTT), and cable networks in France. For these channels, advertising slots are not sold one by one but as packages, also known as plans. For example, a customer (i.e., an advertising company) might ask TF1 to build a gold plan for 30-second commercials on Eurosport during the third week of November, with a budget of €20,000. The key aspect of this offer lies in the audience guarantee attached to the plan. Even if the advertiser does not select advertising slots, it knows that the collective audience will reach a contractual minimum of, for example, 16 GRP (recall that GRP measures the number of viewers of an advertising campaign). More precisely, this transcription of a budget into an audience depends on the selected terms of the package. For example, a gold plan might have 60 percent of GRP scheduled during prime time hours and cost €800 per GRP. For TF1, building such a plan amounts to solving a multidimensional knapsack problem in which a specific number of slots must be selected among the available ones in such a way that the sum of the prices is less than €20,000 and the sum of GRPs is larger than 16, although 60 percent of these GRPs come from prime time slots. Bollapragada et al. (2002) discuss a related optimization problem for the NBC channel.

Optimization and Difficulty

Solving this knapsack problem by hand is a complex and tedious task for media planners, who are responsible for building and selling advertisement plans to advertisers. Hence, we developed a preliminary integer program to automatically compute such plans, based on these general conditions of sales. After a few days of work, we were able to present our first plans using an Excel-based visualization. Prototyping, which is never a waste of time in OR, revealed that the actual problem was far more complex than we had originally believed. Most of our plans were rejected by experts because they presented patterns unacceptable to our customers: commercials in consecutive TV breaks, an unbalanced dispatch of commercials among days of the week, or repeated broadcasting hours on consecutive days (i.e., risk that the commercial will be seen by the same viewers every day). These side constraints were so implicit in the practice of advertising planning that they had been omitted in the specifications. We understood that almost all constraints are soft, that is, a proposal must be tailored for our customer, possibly for a smaller budget or by a relaxation of some dispatching constraints. This would be the case if airing-time inventory is saturated or customer requirements conflict. For example, when launching a new product, the advertiser might specify

that it wants 60 percent of slots on the first days of the plan, which contradicts balancing rules. Finally, optimizing revenue requires minimizing losses resulting from small residual durations in TV breaks: if we assign a 25-second commercial to a 30-second TV break, the residual 5 seconds will be difficult to sell. Using statistics on advertisement durations, this cost can be estimated and taken into account.

Solution Technique

As a result, our resolution scheme is based on goal programming (GP) modeling of these constraints, with an intensive use of classical GP techniques such as lexicographic orders, weighting, satisfaction intervals, min-max, and min-sum (Kornbluth 1973, Ignizio 1983). The majority of instances result in integer programs with less than 1,000 binary decision variables, which are solved in a few seconds by the open-source branch-and-bound solver GLPK 4.24 (Makhorin 2000). However, for larger instances (more than 10,000 binary decision variables), obtaining near-optimal solutions requires several dozen minutes, which is not acceptable to users. Thus, our challenge was to speed up this solution technique, while avoiding the use of a commercial solver (which is helpful in lowering development and maintenance costs). In our case, a main observation is that larger integer programs are difficult to solve because of numerous symmetries. When the number of available TV breaks in a day is large, audiences (and thus prices) do not vary much between two consecutive breaks, resulting in many equivalent branches in the branch-and-bound tree. Because assigning commercials to consecutive TV breaks is undesirable, a good preprocessing heuristic consists of restricting the set of available TV breaks (and thus the search space) by randomly removing one available TV break from two consecutive ones. In the same way, we added additional soft constraints to put lower and upper bounds on the number of commercials planned; this makes sense from a business perspective. Ultimately, the resulting integer programs are all solved in less than 1 minute on a standard computer using GLPK 4.24 (Makhorin 2000). TF1 paid €70,000 to develop this OR engine. The sales front office uses this software daily to build all the plans for the 15 theme channels that the group manages, generating a total revenue of €150 million per year.

Back-Office Software: Optimizing the Revenue of TF1

Business Context

The French TV advertising market is based on sales openings; every other month, the main channels open their reservation planning for 60 days (approximately 5,000 commercial breaks, each with a different price). Advertising companies then send their requests to their favorite channel. For example, Dior might send the following request to TF1: “I want to buy 30 seconds in the 8:40

pm commercial break on May 24th for my Miss Dior perfume." TF1 receives up to 50,000 such requests. One or two weeks after receiving a request, TF1 returns its acceptance decision for that request. Each advertising spot can be cancelled or moved until a few days before its broadcasting date, new bookings can be made, and late buying is possible under different conditions. We note that there is no auction mechanism because prices are fixed by TF1; however, they can depend on the product sector. Although the last stage of this process is a classical yield management situation similar to airline reservations or hotel bookings, the opening stage offers a unique opportunity to perform a global revenue optimization. Making the right decisions during this stage is crucial because 60 to 70 percent of airing time sold currently is eventually broadcast without modification or cancellation (and the received requests generally exceed the total air time for sale).

Optimization and Difficulty

For each commercial break, the problem can be seen as a multidimensional knapsack problem. We want to select a maximal price subset of requests for this commercial break. The sum of durations cannot exceed the available duration; at most, one product is accepted per commercial sector (e.g., cars, perfume, toys) and six spots have short durations (less than 11 seconds). In essence, the economic function to be maximized is the total revenue, that is, the sum of the prices of all accepted requests. However, this objective ignores the importance of customer satisfaction (customers are the advertising companies that contract with TF1). Indeed, even if refusing all requests from Coca-Cola but accepting all requests from Pepsi would yield optimal revenue for this period, an unhappy Coca-Cola could decide to severely decrease its requests for the next opening. Ensuring a certain acceptance rate for each customer is thus a priority (the acceptance rate is the price of accepted requests for this customer divided by the total price of its requests); that is, the first euros accepted in each budget are more valuable than the last. In our case, the objective is to achieve the best possible equity, which means that increasing the least-accepted budget by €1 is always preferable to increasing any other budget by €1. Technically speaking, it means that the objective to be maximized is not directly the sum of the accepted budgets for each customer, but rather the sum of a concave utility function of each customer's accepted budget. With this equity constraint the problem is no longer separable: each knapsack (commercial break) cannot be solved independently from the others. In this context, a natural heuristic proceeds as follows:

1. Let C be the set of advertising companies;
2. **While** C is not empty **do**
   3. Select c in C with minimum acceptance rate;
   4. **If** c has no acceptable request anymore, **then** remove c from C;

5. **Else,** accept one of the acceptable requests from c.

Naturally, the revenue obtained with this procedure is smaller than the sum of optimal revenues for each knapsack. If the request accepted in step 5 is picked at random, the gap is nearly 4 percent. It represents the cost of the equity constraint. Although it might seem to have a small impact, the revenue of a two-month period amounts to hundreds of millions of euros. We will next show that this cost can be dramatically reduced by extracting sensitivity information from dynamic programs.

Solution Technique

The multidimensional knapsack for each commercial break can be modeled as a dynamic program. Within this model, an optimal packing for a commercial break is an optimal path in the layered graph representing the dynamic program. An interesting byproduct of the computation of this optimal path is that it gives the value of the optimal path from each node of this graph to the terminal node (i.e., the Bellman values). If we perform this computation twice (from left to right and from right to left), we obtain for each node the value of the optimal path from the initial node and to the terminal node—namely, the value of the optimal path crossing this node. Actually, considering all arcs associated with the acceptance of a specific request, we can extract the values of both the best packing containing this spot and the best packing excluding this spot (see the appendix for details). The difference between the former and the latter is called the regret for this request (i.e., the loss resulting from the refusal of this request). This regret is positive if and only if the request belongs to an optimal packing for this break and strictly positive if this optimum is unique.

Let us consider a 60-second commercial break for which we have four requests for durations of 20, 20, 30, and 30 seconds, each from a different sector and each paying the same price of €1,000 per second. The best packing with a 30-second spot is €30,000+€30,000; without one of the 30-second spots, the best possible revenue is €30,000+€20,000. Hence, the regret for 30-second requests is €60,000−€50,000 (i.e., +€10,000). A similar reasoning for 20-second requests leads to a regret of −€10,000. Computing regrets for all requests merely requires solving each dynamic program twice and makes it possible to replace the random selection in step 5 of our algorithm by the choice of the request with maximum regret among those of the selected company. Intuitively, it means that rather than accepting a random spot, we pick a spot that fits especially well with other requests for the same commercial break, or at least results in the smallest possible loss on its break (when all regrets for the selected company are negative).

Of course, each time a spot is accepted, the regrets for requests on the same commercial break must be recomputed. Assume that we have a third 20-second request in our previous example. In this case, regrets are initially 0 for each of the five requests because each belongs to one of the two optimal packings: 30+30 or 20+20+20. For example, if a 20-second spot is accepted, the situation changes immediately. In the residual dynamic program, the 30-second spots have a negative regret (−€10,000) and the 20-second spots have a positive regret (+€10,000). Finally, the number of dynamic programs to be computed cannot exceed the number of requests plus the number of commercial breaks. In practice, our Java implementation of this algorithm runs in less than three minutes. On average, we reach total revenue that is only 0.2 percent smaller than we would obtain without taking customer satisfaction (equity) into account.

Beyond the optimization performance, we would like to emphasize the robustness and maintainability of this approach. Not only does our algorithm outperform a previous approach based on local search with ejection chains, it is also much easier to maintain because of its greedy behavior. When a spot is accepted, this decision is never reconsidered; that is, if a user notices a questionable packing in a commercial break, the filling of this break can be easily traced and analyzed. The separation between a basic main loop and a sophisticated oracle (i.e., the dynamic programs providing sensitivity information) also proved useful. For the sake of readability, we simplified the problem we present in this paper. A real application is more complex; for example, the utility function is not strictly concave but becomes linear beyond a certain acceptance threshold, products can have multiple sectors, a sector might appear twice in a commercial break under certain conditions, some sectors have a priority during specific time slots, or an additional price could be paid to get the first or last position of a commercial break. New offers and prices are also created every year or for special events such as the World Cup. A large part of the general conditions of sale must be formalized as constraints in the model of the basic main loop so that for each break, the returned set of requests respects all priorities and compatibility constraints. On the contrary, the oracle can approximate or even omit some technical details without seriously affecting the final revenue. This simplicity and maintainability in an evolving context are representative of the qualities taken into account when choosing an algorithm for OR.

Lessons Learned

As the above examples illustrate, our activity ranges from consulting studies to critical front-office software. We conducted similar revenue optimization projects for most aspects of the sales process, including late buying orders, preferred-positions dispatch, cancellation forecasts, and yield management strategies. These systems allow TF1 to react better and more quickly to customer requests

than its competitors can, thus reinforcing its leading position in the TV advertising market. They also contribute to optimizing the usage of its limited available airtime inventory. The management of TF1 advertising (*TF1 Publicité*) estimates the resulting increase of revenue for TF1 at €20 million per year.

In this final section, we present the principles that we successfully applied in this mission and on other projects conducted both within the Bouygues group and outside of it (e.g., inventory routing, construction-site planning, road maintenance, contact center scheduling).

Modeling and Solving

From an algorithmic point of view, we try not to use a single preferred technique; instead, we use mathematical programming (e.g., linear and integer programming, Lagrangian relaxation), constraint programming, local search, dynamic programming, or dedicated constructive algorithms, depending on the problem. We also expect our staff to have good programming skills because we are convinced that materializing our analysis into operational software is an essential part of our job. The conjunction of these two requirements makes us a team of PhD programmers. When specifying a problem, we are reluctant to add hard constraints to a model, except when expressing physical limitations (e.g., the same channel cannot simultaneously broadcast two messages). More precisely, we try to ensure that finding a feasible solution is easy. The worst situation for an OR system is when the end user sees a message box indicating that no solution has been found. For example, in our third example (sales opening), refusing all requests is a feasible solution, because the minimum acceptance rate for each advertiser is not modeled as a constraint but as a first-rank objective (i.e., minimization of the sum of distances to minimum rates). Even when a solution violating such soft constraints is unacceptable, seeing it is very useful to the user in detecting the cause of the problem, which is almost always an inconsistency in input data.

Project Management

Presenting solutions to the client is the best way of identifying data inconsistencies or specification lapses. To address these issues as early as possible during our projects, we tend to schedule an early beta delivery in our project planning. This beta version usually requires a few weeks of work and consists of an algorithm (generally greedy) that returns a valid solution; however, it might exceed the expected execution time or omit a few constraints. The goal of this release is twofold. First, it requires a set of input data from the client to verify that real data satisfy the preconditions stated in the specifications. Scheduling this task early is crucial because when inconsistencies are identified, fixing them can be time-consuming for the client and should be started as soon as possible. It is

also a prerequisite for the development of the actual optimization algorithm. An efficient way to stress the importance of data in an OR project is to include in the contract this necessary delay between the delivery of valid data sets by the client and the delivery of the software by the vendor. Arguably, it is the only way. Our advertising plan example illustrates the second advantage of this beta release. Presenting first solutions often leads to specifications being refined. It can also demonstrate the feasibility of the project and its potential return on investment. We have seen cases in which end users doubted that an automatic system could produce useful solutions and thus repeatedly postponed change management tasks until the first solutions presented restored their confidence.

Software Integration

We cannot emphasize enough the importance of data management and IT in real-world OR applications. Our team has always included an IT expert who can plug our algorithms into any information system. After 15 years of experience, we know that this role is vital to our business. Prior to delivering the beta version we mentioned in the previous paragraph, we always deliver an empty shell or alpha version to ensure that communications between our algorithm and the host application work well. We can usually deliver it a few days after the approval of detailed application programming interfaces. For most of our applications, we work in partnership with our client's IT department. For example, in the front-office application described above in the *Front-Office Software: Optimizing Advertisers' Plans* section, TF1 developed the user interfaces and interactions with the information system and we (the corporate lab) implemented the optimization algorithm, using the C# 2.0 language and GLPK as native ISO C library. In this way, the integration of our optimization software into TF1's information system, based on the Microsoft .NET framework, was simply done by delivering two Microsoft Windows dynamic link libraries (DLLs). Our experience on various applications within the Bouygues group shows that although bridges between heterogeneous programming frameworks (e.g., Java and .NET) are always possible, using our customer's environment is nevertheless the safest choice.

Research

In conclusion, we would like to point out that our OR team acts both as a research center and a profit center. Despite our corporate positioning, our subsidiaries pay for our services as if we were an external company. This mechanism ensures that we spend the right amount of time on the right projects. Let us consider the three examples described in this paper. Internet sales are approximately 10 times smaller than thematic channel revenues, which are in turn 10 times smaller

than the income of our general interest channel (€1.4 billion). Not surprisingly, the costs of these three projects were consistent with these stakes.

However, we devote approximately 30 percent of our time to academic research. This research activity sometimes consists of scientific developments on operational projects, considering special cases, lower bounds, or alternative solution approaches. The IT culture of the team also leads us to develop optimization software; examples include the constraint programming solver, Choco (Laburthe 2000) and an innovative black-box local-search solver for 0-1 programming, LocalSolver (Benoist et al. 2011). Our scientists can also use this research time to solve academic problems. This presence in the academic community keeps our technical knowledge current or at least gives us an invaluable network of scholars to draw upon when we face a problem outside of our field of expertise. This is how we found the state-of-the-art UCB algorithm for our bandit problem as discussed in the *Mathematical Consulting: Optimizing Internet Click Rates* section. This academic facet of our work also attracts high-caliber candidates when we have a job or intern position available. In practice, some of the optimization engines that we deliver to our customers are a direct result of this research activity. For example, the ordering of advertising messages within a commercial break is optimized with the above-mentioned LocalSolver. Even apparently useless research can turn out to be extremely profitable: the algorithm described in the *Back-Office Software: Optimizing the Revenue of TF1* section was directly inspired by the technique that Benoist et al. (2001) developed for a sport scheduling problem.

Acknowledgments

We would like to thank all the people who have developed and maintained the partnership between TF1 Publicité and the Bouygues e-lab during the past 15 years. They include Éric Bourreau, Yves Caseau, Jacques Deregnaucourt, Étienne Gaudin, Emmanuel Guyot, François Laburthe, Hugues Laigneau, Bruno Martin, Jacques Masson, and Benoît Rottembourg. We also express our gratitude to the anonymous referees for their meticulous reviewing, which has resulted in an improved paper.

Appendix. Sensitivity Analysis on Commercial Breaks

Consider a commercial break of duration D, for which K requests have been made whose durations and prices are $d_1, d_2, \ldots, d_K$ and $p_1, p_2, \ldots, p_K$, respectively. We assume that all products have distinct commercial sectors and all messages have a duration longer than 11 seconds; that is, only the duration constraint applies. However the following reasoning remains valid for the dynamic program that models all three constraints (duration, sectors, and small sizes). Classically, we can define backward$[d, k]$ as the maximum possible revenue

that can be obtained by selecting requests from 1 to k, for a total duration shorter than d. With this formalism, the maximum revenue for this commercial break is backward$[D, K]$ and can be computed with the following recursive formula: backward$[d, k] = \max(\text{backward}[d, k-1], \text{backward}[d - d_k, k-1] + p_k)$. Symmetrically, we can define forward$[d, k]$ as the maximum possible revenue that can be obtained by selecting requests from $k+1$ to K, for a total duration shorter than $D - d$; therefore, we have this equation: forward$[d, k] = \max(\text{forward}[d, k+1], \text{forward}[d + d_k, k+1] + p_k)$. The best possible possible revenue among all selections, including request k, is directly available as

$$\max_{d \in [0, D-d_k]} (\text{backward}[k-1, d] + p_k + \text{forward}[k+1, d+d_k]),$$

while the best possible revenue among all selections, *not* including request k, is directly available as

$$\max_{d \in [0, D]} (\text{backward}[k-1, d] + \text{forward}[k+1, d]).$$

Finally, with only twice the complexity of a classical dynamic programming approach, we obtain sensitivity information for each request on this commercial break.

References

Auer, P., N. Cesa-Bianchi, P. Fisher. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine Learning* **47**(2–3) 235–256.

Benoist, T., B. Estellon, F. Gardi, R. Megel, K. Nouioua. 2011. LocalSolver 1.x: A black-box local-search solver for 0-1 programming. *4OR: Quart. J. Oper. Res.* **9**(3) 299–316.

Benoist, T., F. Laburthe, B. Rottembourg. 2001. Lagrange relaxation and constraint programming collaborative schemes for travelling tournament problems. *Proc. 3rd Internat. Workshop Integration AI OR Techniques Constraint Programming Combin. Optim. Problems (CPAIOR 2001)*. Ashford, UK.

Bollapragada, S., H. Cheng, M. Phillips, M. Garbiras, M. Scholes, T. Gibbs, M. Humphreville. 2002. NBC's optimization systems increase revenues and productivity. *Interfaces* **32**(1) 47–60.

Gabszewicz, J., D. Laussel, N. Sonnac. 1999. TV-broadcasting competition and advertising. Technical Report 99-72, Center for Operations Research and Econometrics (CORE), Université Catholique de Louvain, Louvian, Belgium.

Goettler, R.L., R. Shachar. 2001. Spatial competition in the network television industry. *RAND J. Econom.* **32**(4) 624–656.

Ignizio, J. P. 1983. Generalized goal programming: An overview. *Comput. Oper. Res.* **10**(4) 277–289.

Kornbluth, J. 1973. A survey of goal programming. *Omega* **1**(2) 193–205.

Laburthe, F. 2000. Choco: Implementing a CP kernel. *Proc. Post-Conference Workshop CP 2000 Techniques Implementing Constraint Programming Systems (TRICS 2000)*. Singapore.

Makhorin, A. 2000. GLPK: GNU linear programming kit. Accessed November 23, 2011, `http://www.gnu.org/software/glpk/`.

Nakamura, A., N. Abe. 2005. Improvements to the linear programming based scheduling of web advertisements. *Electronic Commerce Res.* **5**(1) 75–98.

Pandey, S., C. Olston. 2007. Handling advertisements of unknown quality in search advertising. B. Schölkopf, J. Platt, T. Hoffman, eds., *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 1065–1072.